

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平7-181967

(43) 公開日 平成7年(1995)7月21日

(51) Int. Cl. <sup>6</sup>	識別記号	庁内整理番号	F I	技術表示箇所
G 1 0 H 1/00	1 0 1 C			
	Z			
7/02		8938-5H	G 1 0 H 7/ 00	5 2 1 F
審査請求 未請求 請求項の数2 O L (全 16 頁)				

(21) 出願番号 特願平5-328841

(22) 出願日 平成5年(1993)12月24日

(71) 出願人 000116068

ローランド株式会社

大阪府大阪市北区堂島浜1丁目4番16号

(72) 発明者 佐藤 健二

大阪市北区堂島浜1丁目4番16号 ローランド株式会社内

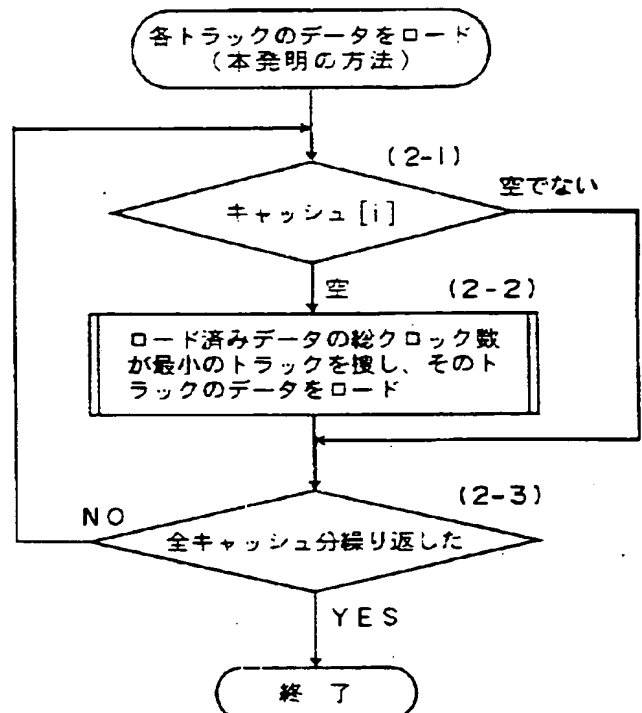
(74) 代理人 弁理士 山田 正紀 (外2名)

(54) 【発明の名称】 演奏データ読込装置

(57) 【要約】

【目的】 本発明は、例えばフロッピーディスク等の外部メモリに記憶された演奏データを読み込む演奏データ読込装置に関し、内部メモリの使用効率を向上させる。

【構成】 空いている（または空いた）キャッシュにどこかのトラックの演奏データをロードするかを固定的には定めず、ロードされている演奏データの総クロック（演奏時間）の最小のトラックの演奏データをロードする。



## 【特許請求の範囲】

【請求項1】 装填された外部メモリから、1つの曲を構成する複数のトラックの演奏データを読み込む演奏データ読込装置において、

前記外部メモリから読み込まれた演奏データを、該外部メモリをアクセスする単位で格納する複数の内部メモリと、

前記複数の内部メモリに格納された演奏データを、各トラックそれぞれについて順次取り出すデータ取出し手段と、

前記複数の内部メモリ中に空の内部メモリが発生した際、前記複数の内部メモリに格納されている各トラック毎の演奏データのうち最短時間で演奏が終了するトラックの演奏データに続く、該トラックの次の演奏データを前記外部メモリから読み込んで空の内部メモリに格納する読込制御手段とを備えたことを特徴とする演奏データ読込装置。

【請求項2】 ディスク装置から、1つの曲を構成する複数のトラックの演奏データを読み込む演奏データ読込装置において、

前記ディスク装置から読み込まれた演奏データを、該ディスク装置をアクセスする単位で格納する複数の内部メモリと、

前記複数の内部メモリに格納された演奏データを、各トラックそれぞれについて順次取り出すデータ取出し手段と、

前記複数の内部メモリ中に空の内部メモリが発生した際、前記複数の内部メモリに格納されている演奏データの各トラック毎の演奏終了時刻と、前記ディスク装置の現在のヘッド位置とに基づいて、前記ディスク装置の、次にアクセスされるメモリ領域を定め、該メモリ領域に記憶された演奏データを前記ディスク装置から読み込んで空の内部メモリに格納する読込制御手段とを備えたことを特徴とする演奏データ読込装置。

## 【発明の詳細な説明】

## 【0001】

【産業上の利用分野】本発明は、例えばフロッピーディスク等の外部メモリに記憶された演奏データを読み込む演奏データ読込装置に関する。

## 【0002】

【従来の技術】近年、自動演奏装置が広く使用されている。自動演奏装置は、通常、外部メモリ、例えばフロッピーディスク（以下フロッピーディスクで代表させる）が装填可能に構成されており、好みの楽曲を表わす演奏データが記憶されたフロッピーディスクを装填して演奏を楽しむことができるように構成されている。

【0003】このように構成された自動演奏装置において、自動演奏を行なうには、従来は装填されたフロッピーディスクに記憶された演奏データを一旦RAM等の内部メモリに一曲分全部を転送し、その転送の後、その内

部メモリから演奏データを順次読み出して演奏を行なっていた。ところがこの方法では少なくとも一曲分の演奏データを記憶できる程度の大容量の内部メモリが必要であり、近年一曲分の演奏データがますます増加する傾向にあるため、ますます大容量の内部メモリを備える必要があり、コスト上好ましくない傾向となっている。

【0004】これを解決するため、以下に説明する手法が提案されている。図27は、従来の手法における演奏データのロード方法を示すフローチャート、図28はその従来の手法の説明図である。ここでは、1つの曲を構成するのに必要な演奏の各パート毎の演奏データ、例えばリズムを表わす演奏データ、メロディを表わす演奏データ、ベースを表わす演奏データ等のそれぞれを「トラック」と称する。またここでは、フロッピーディスクのアクセスの単位を「セクタ」と称する。

【0005】図28に示すように、この自動演奏装置では最大5トラックの演奏を行なうことができるものとし、各トラック毎に、セクタ単位のデータが格納される一時メモリであるキャッシュを、キャッシュAとキャッシュBの2つ用意する。そして、図27のステップ27\_1に示すように、各トラック毎のキャッシュA、Bが空いているかどうか判断し、空いているキャッシュがあればそのキャッシュに演奏データをロードし（ステップ27\_2）、それを全トラックについて順次繰り返していた（ステップ27\_3）。

【0006】例えばトラック0～4について、各トラック専用の各2つのキャッシュA、Bを用意し、それらのキャッシュA、Bにデータを格納し、キャッシュAのデータを1パケット単位で読み出してバッファに一旦格納し、そのバッファからクロックに応じて1メッセージ単位のデータとしてMIDI出力する。ここで、例えばトラック0についてキャッシュAの演奏データの読み出しが終了すると、それに引き続いてキャッシュBの演奏データが1パケット単位で順次バッファに転送され、その間にキャッシュAに、キャッシュBに格納された演奏データに引き続く演奏データがフロッピーディスクから読み出されて格納される。以後同様にして、キャッシュAとキャッシュBが交互に用いられる。他のトラック2～5についても同様である。各トラック毎にキャッシュを2つ用意するのは、フロッピーディスクからキャッシュに演奏データを転送するのに、1回の転送あたり例えば0.25秒程度必要とし、これはテンポ120で考えると8分音符1つ分程度に相当し、1つのトラックについてキャッシュを1つしか備えないとそのキャッシュに格納された演奏データによる演奏が終了した後次の演奏データが転送されるまでの間、演奏が中断してしまうからである。

【0007】このように構成することにより、フロッピーディスクに格納された演奏データを1曲分まとめてロードすることのできる大容量の内部メモリを備えること

なく、演奏データをフロッピーディスクから少しずつ順次ロードしながら演奏を行なうことができる。

#### 【0008】

【発明が解決しようとする課題】上述したように1回のデータ転送に約0.25秒程度必要とするため、上述の、フロッピーディスクから内部のキャッシュに少しずつデータ転送を行なう手法では、この転送速度よりキャッシュからデータを送出する速度の方が速い、例えばテンポの速い演奏を行う場合、クロックに正確な安定した演奏を行なうためには、1つのトラックあたりのキャッシュの数を増やす必要がある。しかし、そのように1つのトラックあたりのキャッシュの数を増やすと、キャッシュとして必要な全メモリ容量はその装置の最大トラック数に比例して膨大なものになってしまうことになる。また曲目によっては、その装置の最大トラック数よりも少ない数のトラックしか使用しない場合もあり、そのときにはその曲の演奏の間ずっと空のままのキャッシュが生じてしまい、メモリの使用効率が悪いという問題がある。

【0009】本発明は、上記事情に鑑み、内部メモリの使用効率を向上させた演奏データ読込装置を提供することを目的とする。

#### 【0010】

【課題を解決するための手段】上記目的を達成する本発明の第1の演奏データ読込装置は、装填された外部メモリから、1つの曲を構成する複数のトラックの演奏データを読み込む演奏データ読込装置において、

(1) 外部メモリから読み込まれた演奏データを、外部メモリをアクセスする単位で格納する複数の内部メモリ

(2) 上記複数の内部メモリに格納された演奏データを、各トラックそれぞれについて順次取り出すデータ取出し手段

(3) 上記複数の内部メモリ中に空の内部メモリが発生した際、それら複数の内部メモリに格納されている各トラック毎の演奏データのうち最短時間で演奏が終了するトラックの演奏データに続く、そのトラックの次の演奏データを外部メモリから読み込んで空の内部メモリに格納する読込制御手段

を備えたことを特徴とする。

【0011】また、上記目的を達成する本発明の第2の演奏データ読込装置は、上記第1の演奏データ読込装置にいう外部メモリとしてディスク装置を備え、さらに、上記(3)の読込制御手段に代えて、

(4) 上記複数の内部メモリ中に空の内部メモリが発生した際、それら複数の内部メモリに格納されている演奏データの各トラック毎の演奏終了時刻と、ディスク装置の現在のヘッド位置とに基づいて、ディスク装置の、次にアクセスされるメモリ領域を定め、そのメモリ領域に記憶された演奏データをディスク装置から読み込んで空の内部メモリに格納する読込制御手段

を備えたことを特徴とする。

#### 【0012】

【作用】複数のトラックの演奏データをよく観察してみると、テンポトラックのようにほとんどデータがなかったり、リズムトラックやギターデータが記録されているトラックのように大量のデータが記録されていたりと、データ量は同一ではない。

【0013】そこで、本発明のうち第1の自動演奏装置では、空いている（または空いた）内部メモリにどのトラックのデータをロードするかを固定的には定めず、ロードされているデータの総クロック最小（演奏時間最短）のトラックのデータをロードするようにしたため、内部メモリの利用効率が上がり、内部メモリの数が少なめであっても、過密データの、クロックに正確な演奏が可能になる。

【0014】また、外部メモリとして、ディスク装置を備えた場合に、そのディスク装置に記憶されたデータをロードする際に、そのディスク装置のヘッドを大きく移動させるとそれだけロードする迄に時間がかかり、したがってヘッドの動きの少ない順序でロードすることが好ましい。本発明の第2の自動演奏装置は、この観点から成されたものであり、基本的には上記第1の自動演奏装置と同様ではあるが、次にロードすべきデータを、ロードされているデータの総クロックの最小トラックのデータに固定するものではなく、総クロックの小さいいくつかのデータの中からヘッドの動きが少なく済むデータを先にロードすることにより、全体としてより迅速なロードが行なわれる。

#### 【0015】

【実施例】以下、本発明の実施例について説明する。図1は、本発明の演奏データ読込装置の一実施例の構成を表わすブロック図である。この自動演奏装置10には、プログラムを実行するCPU11、CPU11で実行されるプログラムを格納するROM12、プログラムの作業領域として使用されるRAM13が備えられている。本実施例では、このRAM13内部に、本発明にいう複数のキャッシュが確保される。

【0016】またこの自動演奏装置10には、演奏の開始（PLAY）、停止（STOP）等を指示するスイッチ14、この自動演奏装置10の状態、例えば演奏中の曲の曲番号や小節番号等を表示する表示器15、フロッピーディスク装置16を制御してフロッピーディスク装置16に装填されたフロッピーディスクをアクセスするフロッピーディスク制御装置17、演奏データをMIDI出力端子から出力するMIDI通信装置18が接続されている。これらの各構成要素11～15、17、18はバス19により互いに接続されている。MIDI通信装置18から出力された演奏データは、外部に接続された。例えば自動演奏装置に入力され、この自動演奏装置により、入力された演奏データに基づく自動演奏が行な

われる。尚、ここでは、演奏データ読込装置に自動演奏装置が接続されるものとして説明するが、本実施例の演奏データ読込装置を内包した自動演奏装置を構成してもよいことはもちろんである。

【0017】図2、図3は、本実施例におけるデータ転送処理の基本概念を示した、それぞれフローチャート、データの流れを示す説明図である。図3に示すように、この実施例では5つのキャッシュが用意されており、図2のルーチンのステップ2\_\_1において番号*i* ( $i=0\sim4$ )のキャッシュが空か否かを判断し、空のキャッシュがあった場合、ステップ2\_\_2において、ロード済データの総クロック数が最小のトラックを捜し、そのトラックのデータをロードする。これを全キャッシュについて繰り返す(ステップ2\_\_3)。すなわち、ここには、キャッシュとトラックとの固定的な対応づけはなく、空きのキャッシュが発生する毎に、総クロック数が最小、即ちロードした演奏データが最も早くなくなってしまうトラックの演奏データを、その空のキャッシュに転送するようにしたものである。

【0018】以下、このデータ転送について詳細に説明する。図4は、本実施例における、フロッピディスク装置16(図1参照)に装填されたフロッピディスク中の、今から演奏しようとするある曲の演奏データのデータ構造を示した図である。この曲の演奏には3つのトラックが使用され、トラック1の演奏データは、そのフロッピディスクのセクタ0と、セクタ1の一部とに記憶されており、その各セクタ内のトラック0の演奏データのクロック数(演奏時間)はそれぞれ150、10である。またトラック1の演奏データは、セクタ1の一部と、セクタ2、セクタ3と、セクタ4の一部とに記憶されており、その各セクタ内のトラック1のクロック数は、それぞれ、10、50、50、50である。同様に、トラック2の演奏データは、セクタ4の一部と、セクタ5、6に記憶されており、各セクタ内のトラック2のクロック数は、それぞれ20、100、40である。

【0019】図5～図14は、フロッピディスクに記憶された図4に示すデータ構造の演奏データが、図3に示す5つのキャッシュに転送(ロード)される際の転送手順と、そのデータ転送に伴う図1に示すCPU11で実行されるデータ転送のためのルーチンで利用される各種フラグ、レジスタ等の値の変化を示した模式図である。

【0020】これらの図5～図14に示す各種フラグ、レジスタ等の役割りは以下のとおりである。

・LoadPtr[i] : 番号*i* ( $i=0\sim2$ )のトラックの演奏データを最後にロードしたキャッシュの番号( $0\sim4$ )を格納するレジスタ

・LoadTime[i] : 番号*i*のトラックの、それまでキャッシュにロードされた演奏データの総クロック数を格納するレジスタ

・SectCnt[i] : 番号*i*のトラックの、未だキャッシュにロードされていない、残りのセクタの数を格納するレジスタ

・PlayNo[i] : 番号*i*のトラックの、次に出力される演奏データが格納されているキャッシュの番号を格納するレジスタ

・Cache[j] : 番号*j*のキャッシュにロードされた演奏データが記憶されていた、フロッピディスクのセクタの番号を格納するレジスタ

・Free[j] : 番号*j*のキャッシュが空('1')であるか否か('0')を示すフラグ

・Next[j] : 番号*j*のキャッシュに格納された演奏データに引き続く、その演奏データのトラックと同一のトラックの演奏データが格納されたキャッシュの番号を格納するレジスタ。自分が最後のときは'-1'が格納される。

【0021】以下、図5～図14について順を追って説明する。図5、図6は、演奏開始前の演奏準備段階におけるデータ転送を示すものであり、先ず図5に示すように、最初は全てのキャッシュが空であるから、キャッシュ0、1、2に、それぞれトラック0、1、2の各先頭の演奏データが格納されたセクタ0、1、4の演奏データが順次転送される。これにより、LoadPtr[0]、LoadPtr[1]、LoadPtr[2]に、それぞれ、転送先のキャッシュ番号0、1、2が格納され、LoadTime[0]、LoadTime[1]、LoadTime[2]に、それぞれ、転送されたセクタ内の各トラック0、1、2の演奏データの総クロック数'150'、'10'、'20'が格納される。またSectCnt[0]、SectCnt[1]、SectCnt[2]には、それぞれ、トラック0、1、2の演奏データが格納された、未だキャッシュに転送されていないセクタ数'1'、'3'、'2'が格納される。

【0022】さらにPlayNo[0]、PlayNo[1]、PlayNo[2]には、演奏のための各トラックの演奏データの取り出し先であるキャッシュ番号0'、'1'、'2'が格納される。またCache[0]、Cache[1]、Cache[2]には、そのキャッシュに転送された演奏データが記憶されていたフロッピディスクのセクタ番号'0'、'1'、'4'が格納される。キャッシュ3、4には未だ転送されていないため、Cache[3]、Cache[4]は空である。またキャッシュ0、1、2にはデータが転送されたため、Free[0]～Free[2]には、それぞれ'0'が格納される。キャッシュ3、4は空であるためFree[3]、Free[4]には'1'が格納されている。さらに、キャッシュ1、2、3に格納された演奏データが、現在のところ、各トラック0、1、2について転送された演奏データの最終であるた

め、Next [0] ~ Next [2] には、最終であることを示す '−1' が格納され、また、Next [3], Next [4] には初期値としての '−1' が格納されている。

【0023】次に、LoadTime [0] ~ LoadTime [2] を比較し、総クロック数の最小のトラック（ここではトラック1）について空のキャッシュ3に演奏データが転送され、空のキャッシュ4がまだ残っているため、キャッシュ3にデータが転送された段階における、LoadTime [0] ~ LoadTime [2] を比較し、総クロック数の最小のトラック（今度はトラック2）について空のキャッシュ4に演奏データが転送され、これにより、キャッシュ0~4の全てについて演奏データが転送される。

【0024】図6は、そのときの状態を示した図である。LoadPtr [1], LoadPtr [2] には、それぞれ、対応するトラックの、最後に転送したデータのデータ転送先であるキャッシュ番号 '3', '4' が格納され、LoadTime [1], LoadTime [2] には、それぞれ、トラック1, 2に関する転送された演奏データの総クロック数 '10+50', '20+120' が格納され、SectCnt [1], SectCnt [2] には、それぞれ各トラック1, 2のデータ未転送のセクタ数 '2', '1' が格納される。

【0025】また、Cache [3], Cache [4] には、そこに格納された演奏データのセクタ番号 '2', '5' が格納され、Free [3], Free [4] にはデータが格納されていることを示す '0' が格納される。さらに、Next [1], Next [2] には、次の演奏データの格納先であるキャッシュ番号 '3', '4' が格納される。

【0026】このようにして、全てのキャッシュ0~4に演奏データがロードされた後、演奏が開始される。図7は、演奏開始後、10クロックに相当する時間が経過した後の状態を表わすものである。10クロック経過したことにより、LoadTime [0], LoadTime [1], LoadTime [2] は、図6の状態のときよりもそれぞれ10クロックずつ減り、それぞれ '140', '0+50', '10+100' となる。このとき、キャッシュ1に格納されていた演奏データによる演奏が終了し、Free [1] = 1, Next [1] = −1に変更され、PlayNo [1] = 3に変更される。

【0027】これによりキャッシュ1は空になったため、SectCnt [0] ~ SectCnt [2] が '0' ではないトラック（ここではトラック0~2のいずれも '0' ではない）に関し、LoadTime [0] ~ LoadTime [2] の内容が比較され、総クロック数の最小のトラック1の、次の演奏データがキ

ャッシュ1にロードされる。

【0028】図8は、キャッシュ1にトラック1の演奏データがロードされた後の状態を示した図である。キャッシュ1にセクタ3の演奏データが転送され、Cache [1] = 3, Free [1] = 0に変更され、Next [3] = 1に変更され、さらにLoadPtr [1] = 1, LoadTime [1] = 50+50, SectCnt [1] = 1に変更される。

【0029】図9は、図8の状態から10クロック経過後の状態を示す図である。10クロック経過したことにより、LoadTime [0], LoadTime [1], LoadTime [2] は、図8の状態のときよりもそれぞれ10クロックずつ減り、'130', '40+50', '0+100' となる。このときキャッシュ2に格納された演奏データによるトラック2の演奏が終了し、Free [2] = 1, Next [2] = −1に変更される。またPlayNo [2] が4に変更され、キャッシュ4に格納されたトラック2の演奏データによる演奏が開始される。

【0030】また、キャッシュ2は空になったため、SectCnt [0] ~ SectCnt [2] が '0' ではないトラックに関しLoadTime [0] ~ LoadTime [2] の内容が比較され、総クロック数の最小のトラック1の、次の演奏データが、図10に示すように、キャッシュ2にロードされる。キャッシュ2に、トラック1の次の演奏データ、即ちセクタ4の演奏データが転送され、Free [2] = 0に変更され、Next [1] = 2に変更され、さらにLoadPtr [1] = 2, LoadTime [1] = 40+50+50, SectCnt [1] = 0に変更される。

【0031】SectCnt [1] = 0となったことから、トラック1の演奏データは、全てキャッシュにロードされたことになる。図11は、図10に示す状態から、さらに40クロック経過後の状態を示す図である。40クロック経過したことにより、LoadTime [0], LoadTime [1], LoadTime [2] は、図10の状態よりもそれぞれ40クロックずつ減り、'90', '0+50+50', '60' となる。

【0032】このときキャッシュ3に格納されたトラック1の演奏データは空になり、Free [3] = 1, Next [3] = −1に変更され、PlayNo [1] = 1に変更され、今度はキャッシュ1に格納されたトラック1の演奏データによる演奏が行なわれる。またキャッシュ3は空になったため、SectCnt [0] ~ SectCnt [2] が '0' ではないトラックに関し、LoadTime [0] ~ LoadTime [2] の内容が比較され、総クロック数の最小のトラック2の、次の演奏データがキャッシュ3にロードされる。

【0033】図12は、キャッシュ3にトラック2の、

次の演奏データ、即ちセクタ6の演奏データがロードされた状態を示した図である。Cache [3] = 6, Free [3] = 0に変更され、また、Next [4] = 3に変更される。さらに、LoadPtr [2] = 3, LoadTime [2] = 60 + 40, SectCnt [2] = 0に変更される。

【0034】図13は、図12の状態から更に50クロック経過した後の状態を示す図である。50クロック経過したことにより、LoadTime [0], LoadTime [1], LoadTime [2] は、図12に示す状態のときよりもそれぞれ50クロックずつ減少し、それぞれ、40, 0 + 150, 10 + 40となる。

【0035】このとき、キャッシュ1に格納されたトラック1の演奏データは空になり、Free [1] = 1, Next [1] = -1に変更され、PlayNo [1] = 2に変更される。また、キャッシュ1が空になったことから、SectCnt [0] ~ SectCnt [2] が '0' ではないトラックに関し、LoadTime [0] ~ LoadTime [2] が比較されるが、ここではSectCnt [1] = 0, SectCnt [2] = 0、すなわち、トラック1, 2の演奏データのキャッシュへのロードは既に終了しているため、残りのトラック0の演奏データがキャッシュ1にロードされる。

【0036】図14は、キャッシュ1にトラック0の次の演奏データ、即ちセクタ1の演奏データが転送された状態を示した図である。Cache [1] = 1, Free [1] = 0に変更され、またNext [0] = 1に変更される。さらにLoadPtr [0] = 1, LoadTime [0] = 40 + 10, SectCnt [0] = 0に変更される。

【0037】その後10クロック経過すると、LoadTime [0] = 30 + 10, LoadTime [1] = 40, LoadTime [2] = 0 + 40となって、トラック2の、キャッシュ4に格納された演奏データによる演奏が終了し、Free [4] = 1, Next [4] = -1, PlayNo [2] = 3に変更される。さらにその後30クロック経過すると、LoadTime [0] = 0 + 10, LoadTime [1] = 10, LoadTime [2] = 10となってキャッシュ0に格納されたトラック0の演奏データが空になって、Free [0] = 1, Next [0] = -1, PlayNo = 1に変更される。

【0038】さらにその後ろ10クロック経過すると、この曲一曲分の演奏データによる演奏が全て終了する。以下、フロッピディスクに記憶された演奏データのデータフォーマットについて説明し、次いで、図5~図14を参照して説明した演奏データの転送に関連した各種ルーチンについて説明する。

【0039】図15は、フロッピディスクに記憶された演奏データのフォーマットを示した図である。先頭に

1バイトのヘッダがありそこに総トラック数が記述されている。その後は、トラック毎のデータとなっており、トラック毎のデータの先頭の4バイトはそのトラックの長さ(バイト数)が記述され、その後ろに実際の演奏データが記述されている。

【0040】図16は演奏データのフォーマットを示した図である。各演奏データは2バイトの時間情報部分と3バイトのデータ情報部分で構成されている。図17は演奏データ中のデータ情報部分の詳細を示した図である。3バイトのMIDI情報はそのまま記述され、MIDI情報が2バイト、1バイトの場合は余分なところをFFhで埋めている。さらに、時間情報のみ、テンポ、拍子、終了情報なども記述することができる。

【0041】次に、図18~図26のルーチンについて説明するが、既に図5~図14を参照してデータ転送の具体例について詳細に説明したため、以下では、各ルーチンについて簡単に説明するにとどめる。以下に、既出のものも含め、図18~図26のルーチンで使用される主なレジスタ、フラグの役割割りについてまとめて説明する。

・Cache [MAX\_CACHE] [512] : ディスクから演奏データがロードされるキャッシュの番号が格納されるレジスタ

・PlayNo [MAX\_TRACK] : 次に出力される演奏データが格納されたキャッシュの番号が格納されるレジスタ

・PlayAddr [MAX\_TRACK] : 次に出力される演奏データのキャッシュ内のアドレスが格納されるレジスタ

・TimeCount [MAX\_TRACK] : 待時間カウンタ

・LoadPtr [MAX\_TRACK] : 最後にロードされたキャッシュの番号が格納されるレジスタ

・LoadTime [MAX\_TRACK] : 既にキャッシュにロードされた演奏データの、トラック毎のクロック数を記憶するレジスタ

・SectCnt [MAX\_TRACK] : 各トラックのロードが終了したか調べるために演奏データが記録されている残りのセクタ数が格納されるレジスタ

・EndFlag [MAX\_TRACK] : そのトラックの演奏が既に終了したか否かを示すフラグ

・Next [MAX\_CACHE] : 次のキャッシュ番号が格納されるレジスタ

・Free [MAX\_CACHE] : そのキャッシュが空か否かを示すレジスタ ('1' が空を表す)

・SetData [trk] [3] : 出力される演奏データをトラック毎に記憶するレジスタ

・work [5] : 5バイトの演奏データを時間情報部分とデータ情報部分とに分離するために使用される作業用レジスタ

- ・work0:ロード済み演奏データの総クロック数が最小のトラックを捜すために総クロック数を一時的に格納しておく作業用レジスタ
- ・Song:曲番号が格納されるレジスタ
- ・EndCount:曲の終了を調べるためのレジスタ
- ・Buff:総トラック数を調べるための作業用レジスタ
- ・TrkNo:その曲のトラック数が格納されるレジスタ
- ・trk:トラックループカウンタ1
- ・trk0:トラックループカウンタ2
- ・trk1:ロード済クロック (LoadTime) が最小のトラックのトラック番号が格納されるレジスタ
- ・i:ループカウンタ
- ・MeasCount:小節番号計測用レジスタ
- ・Measure:小節番号が格納されるレジスタ

図18は電源オンで実行が開始される、メインルーチンのフローチャートである。

【0042】ステップ18\_1では、フロッピィディスク装置にフロッピィディスクが入っているか否かが調べられる。尚、図中には示されていないが、フロッピィディスクが抜かれると、クロック停止と全音楽消去の処理が行なわれた後、ステップ18\_1に飛ぶ。ステップ18\_2では、曲番号を記憶しておくレジスタSongに'0'がセットされる。本実施例では、簡単のため、曲の演奏は必ず0番の曲から開始されるものとしてステップ18\_3では、初期化が行なわれる。すなわち、全トラック分のPlayNo, PlayAddr, TimeCount, LoadPtr, SectCnt, LoadTimeに'0', EndFlagに'1', 全キャッシュのNextに'0', Freeに'1', MeasCount, Measureに'0'が格納される。

【0043】ステップ18\_4では、曲番号記憶レジスタSongで示される曲の演奏準備が行なわれる。詳細については後述する。ステップ18\_5では、小節カウンタの準備が行なわれる。詳細は後述する。以上で演奏の準備が終了し、ステップ18\_6で、プレイ (PLAY) スイッチが押されるのを待つ。

【0044】PLAYスイッチが押されるとステップ18\_7に進み、クロック割り込みの発生が指示される。ステップ18\_8ではストップ (STOP) スイッチが押されたか否かが調べられ、押されていないときはステップ18\_9に進んで曲の終了を表わすレジスタEndCountが調べられ、曲の途中 (EndCount > 0) のときはステップ18\_10に進む。

【0045】ステップ18\_10では、ディスクからキャッシュ (Cache) へのデータロードが指示され、ステップ18\_11では小節番号表示が指示される。ステップ18\_10, 18\_11の詳細については後述する。一方、STOPスイッチが押された場合、もしくは

曲の終了の場合、ステップ18\_12に進み、STOPスイッチが押されたことに伴う処理が指示される。詳細については後述する。

【0046】ステップ18\_13ではSongに1が加えられ、ステップ18\_4ではSongがフロッピィディスクに記録されている曲数以下か否かが調べられ、フロッピィディスク中にまだ演奏されていない曲が残っているときはそのままステップ18\_3に戻り、最終曲まで演奏されたときはSongに'0'がセットされて先頭の曲に戻った上でステップ18\_3に戻る。

【0047】以上のようにして、フロッピィディスクに記憶された演奏データに基づいて、循環的に演奏が繰り返される。図19は、図18のステップ18\_4で実行される、Songで示される曲の演奏準備を行なうサブルーチンのフローチャートである (図5参照)。先ずステップ19\_1では、その曲のトラック数を調べるために、その曲ファイル (図15参照) の先頭部分の1セクタを、作業用レジスタBuffにロードする。

【0048】ステップ19\_2では、その曲のトラック数を記憶するレジスタTrkNo、曲の終了を調べるためのレジスタEndCountに、ヘッダに書かれているトラック数をセットする。ステップ19\_3では、トラックループカウンタtrkに'0'をセットする。

【0049】ステップ19\_4では、ロードの終了を検出するために、trkトラックで示されるのデータが何セクタに渡って記録されているか計算し、その値をSectCnt [trk] にセットする。ステップ19\_5では、trkで示されるトラックの先頭の演奏データが格納されているセクタの演奏データをCache [trk] にロードする。

【0050】ステップ19\_6では、ステップ19\_5でロードが発生したので、SectCnt [trk] から1が引き算される。ステップ19\_7では、ステップ19\_5で何クロック分の演奏データがロードされたかを調べ、そのクロック数を、キャッシュにこれまでにロードされていた演奏データのクロック数を記憶するレジスタLoadTime [trk] にセットする。

【0051】ステップ19\_8では、最後にロードされたキャッシュの番号を記憶するレジスタLoadPtr [trk] に、trkをセットする。ステップ19\_9では、次に出力される演奏データが格納されたCacheの番号を記憶するレジスタPlayNo [trk] にtrkをセットし、さらに、次に出力される演奏データのキャッシュ内のアドレスを記憶するレジスタPlayAddr [trk] に演奏データの先頭アドレスをセットする。

【0052】ステップ19\_10では、次の演奏データのセットが指示される。詳細については後述する。ステップ19\_11では、trkに'1'が加えられ、ステップ19\_12ではtrkがTrkNo以内か否かが調べ

られ、`trk`が`TrkNo`以内であればステップ19\_4に戻り、`trk`が`TrkNo`を越えるとステップ19\_13へ進む。ステップ19\_13では、図18に示すメインルーチンにおけるステップ18\_10と同様に、演奏データのロードが指示される。ステップ19\_13のルーチンの詳細については後述する。

【0053】図20は、図19のルーチンのステップ19\_10で実行される、次の演奏データをセットする処理を行なうサブルーチンのフローチャートである（図6参照）。ステップ20\_1では、`PlayNo[trk]`が作業レジスタ`no`にセットされ、また、`PlayAddr[trk]`が作業レジスタ`addr`にセットされる。

【0054】ステップ20\_2では、作業用ループカウンタ`i`に‘0’がセットされる。ステップ20\_3では、`i`が5以内か否かが調べられる。これは、演奏データは、図16にあるように5バイトで構成されており、1バイトずつ読み込むため5回読み込む必要があるからである。`i < 5`の場合ステップ20\_4に進み、`addr`が512（1セクタのバイト数）以内か否かが調べられる。`addr ≥ 512`の場合、ステップ20\_5に進み、`PlayNo[trk]`に、次に演奏される演奏データがロードされたキャッシュ番号が記憶されたレジスタ`Next[no]`がセットされ、`PlayAddr[trk]`に‘0’がセットされ、`Next[no]`に、次が存在しないことを示す値‘-1’がセットされ、さらに`Free[no]`にそのキャッシュが空であることを示す値‘1’がセットされる。

【0055】ステップ20\_6では、作業用レジスタ`work[i]`にキャッシュ`Cache[no][addr]`がセットされ、ステップ20\_7で`i`に‘1’が加えられてステップ20\_3に戻る。ステップ20\_3で`i ≥ 5`であると判定されるとステップ20\_8に進み、待時間カウンタ`TimeCount[trk]`に新しい時間が加えられる。

【0056】ステップ20\_9では、出力される演奏データを記憶するレジスタ`SetData`に新しいデータがセットされる。図21は、図18に示すメインルーチンのステップ18\_10、および図19に示すサブルーチンのステップ19\_13で実行される。空のキャッシュをさがしてその空のキャッシュにロード済み時間の最も少ないトラックの演奏データをロードするロードデータサブルーチンのフローチャートである。

【0057】先ずステップ21\_1では、キャッシュの番号を示す作業用カウンタ`i`に‘0’がセットされる。ステップ21\_2では、そのキャッシュが空かどうかを示すレジスタ`Free[i]`が調べられ、空いていればステップ21\_2に進み、空いていなければステップ21\_14に進む。

【0058】以下のステップ21\_3～ステップ21\_

8はロード済み時間の少ないトラックを捜す処理である。ステップ21\_3では、トラック番号を示す作業用レジスタ`trk0`に‘0’がセットされ、ロード済クロックが最小のトラックのトラック番号を示す作業用レジスタ`trk1`に0がセットされ、さらに最小クロック数を捜すための作業用レジスタ`work0`に、本装置で扱える最大時間`MAX_TIME`がセットされる。

【0059】ステップ21\_4では、ロード完了レジスタ`SectCnt[trk0]`が正か否かが調べられ、正であればステップ21\_5に進み、そうでなければステップ21\_7に進む。ステップ21\_5では`work0`が`LoadTime[trk0]`を越えるか否かが調べられ、`work0 > LoadTime[trk0]`の場合ステップ21\_6に進み、そうでないときはステップ21\_7に進む。

【0060】ステップ21\_6では、作業用レジスタ`work0`に`LoadTime[trk0]`がセットされ、さらに`trk1`に`trk0`がセットされる。ステップ21\_7では`trk0`に‘1’が加えられる。ステップ21\_8では、`trk0`が`TrkNo`以下か否かが調べられ、`trk0 < TrkNo`の場合ステップ21\_4に戻り、`trk0 ≥ TrkNo`の場合ステップ21\_9に進む。

【0061】ステップ21\_9では、`work0`がステップ21\_3で格納された`MAX_TIME`から変更されたか調べられ、変更されていないときは終了し、変更されていたときはステップ21\_10に進む。ステップ21\_10以降は、データのロードに伴う処理である。ステップ21\_10では、`trk1`で示されるトラックの、次の演奏データが`Cache[i]`にロードされる。

【0062】ステップ21\_11では、`SectCnt[trk1]`から‘1’が引き算される。ステップ21\_12では、ステップ21\_10で何クロック分の演奏データがロードされたかを調べ、その値が`LoadTime[trk1]`に加算される。ステップ21\_13では、ロードに関するリンク情報の更新のため、最後にロードされたキャッシュの番号を記憶する`LondPtr[trk1]`が`Next[i]`にセットされ、`LondPtr[trk1]`に1がセットされ、`Free[i]`に‘0’がセットされる。

【0063】ステップ21\_14では`i`に‘1’が加えられる。ステップ21\_15では、`i`がキャッシュの数`MAX_CACHE`以内か否かが調べられ、未だ処理の行なわれていないキャッシュがあるときはステップ21\_2に進み、全てのキャッシュについて処理が終了したときはこのルーチンを抜ける。

【0064】図22は、演奏データの出力指示、次のデータのセット、ロード済みカウンタの更新、曲の終りを検出して自動的に演奏終了する指示を行なうクロック割



込みルーチンのフローチャートである。このクロック割込みルーチンは、図18のメインルーチンのステップ18\_7でクロック割込みの発生が指示されたことを受けて発生する割込み用クロックにより、所定時間間隔毎に起動される。

【0065】このルーチンが起動されると、先ずステップ22\_1において、トラックカウンタ`trk`に‘0’がセットされる。ステップ22\_2では、そのトラックの演奏が既に終了しているか否かを示すレジスタ`EndFlag[trk]`が調べられ、終了しているときはステップ22\_8に進み、そうでないときはステップ22\_3に進む。

【0066】ステップ22\_3では、`TimeCount[trk]`から‘1’が引き算される。ステップ22\_4では、`TimeCount[trk]`が正か0以下かが調べられ、正の場合ステップ22\_7に進み、0以下の場合ステップ22\_5に進む。

【0067】ステップ22\_5ではデータの出力が指示される。このステップ22\_5の詳細については後述する。ステップ22\_6では、次のデータのセットが指示される。詳細については、図20が参照され既に説明されている。

【0068】ステップ22\_7では、`LoadTime[trk]`から‘1’が引き算される。ステップ22\_8では、`EndCount`が正か0か調べられ、正の場合ステップ22\_10に進み、0の場合ステップ22\_9に進む。ステップ22\_9では、自動終了のために演奏停止処理が指示される。詳細については後述する。

【0069】ステップ22\_10では`trk`に‘1’が加算される。ステップ22\_11では`trk`が`TrkNo`以下か否かが調べられ、`trk < TrkNo`のときはステップ22\_2に戻り、`trk ≥ TrkNo`のときはステップ22\_12に進む。ステップ22\_12では、小節番号計測用レジスタ`MeasCount`に‘1’が加算される。

【0070】ステップ22\_13では、`MeasCount`が16以上か否かが調べられる。`MeasCount > 16`のときはステップ22\_14に進み、そうでなければそのまま終了する。ステップ22\_14では、`MeasCount`に‘1’がセットされ、小節番号を記憶するレジスタ`Measure`に‘1’が加算される。

【0071】図23は、図22に示すクロック割込みルーチンのステップ22\_5で実行される、演奏データをその種類によりMIDI出力端子に出力したり、テンポ、拍子を更新したり、曲の終りを検出するサブルーチンのフローチャートである。ステップ23\_1では`DataSet[trk]`がMIDI情報であるか否かが調べられ、MIDI情報であった場合はステップ23\_2に進んで、`DataSet[trk]`がMIDI出力端子から出力される。

【0072】ステップ23\_3では、`DataSet[trk]`がテンポ情報であるか否かが調べられ、テンポ情報であった場合はステップ23\_4に進みテンポが変更される。ステップ23\_5では、`DataSet[trk]`が拍子情報であるか否かが調べられ、拍子情報であった場合はステップ23\_6に進み、拍子に変更される。

【0073】ステップ23\_7では`DataSet[trk]`が終了情報であるか否かが調べられ、終了情報であった場合はステップ23\_8に進み、`EndCount`から‘1’が引き算され、`EndFlag[trk]`に‘0’がセットされる。図24は、図18に示すメインルーチンのステップ18\_12で実行される、演奏停止処理を行なうサブルーチンのフローチャートである。

【0074】ステップ24\_1ではクロック割込みが停止され、ステップ24\_2では、発音中の全楽音の消音処理が実行される。図25は、図18に示すメインルーチンのステップ18\_5で実行される、小節カウンタの曲頭の休符区間を検出し、楽譜通りの小節番号にするための処理を行なうサブルーチンのフローチャートである。

【0075】ステップ25\_1では、曲の先頭から最初の発音までの時間（休符時間）が検出され、その値が作業用レジスタ`RestTime`にセットされる。ステップ25\_2では、`RestTime`が3未満か否かが調べられ、3未満であればステップ25\_3に進み、そうでなければステップ25\_5に進む。ステップ25\_3では、`Measure`に‘0’がセットされてこのルーチンを抜ける。

【0076】ステップ25\_4では、`RestTime`が5未満か否かが調べられ、5未満であればステップ25\_5に進み、そうでなければステップ25\_6に進む。ステップ25\_5では、アウフタクトの小節番号として、`Measure`に‘-1’がセットされ、このルーチンを抜ける。ステップ25\_6では、`Measure`に‘0’がセットされ、このルーチンを抜ける。

【0077】図26は、図18のメインルーチンのステップ18\_11で実行される、小節番号を表示する処理を行なうサブルーチンのフローチャートである。ステップ26\_1では、`Measure`が0以上か否かが調べられ、0以上の場合ステップ26\_2に進み、そうでなければ26\_3に進む。ステップ26\_2では、表示器に`Measure+1`の値が表示される。

【0078】ステップ26\_3では、表示器に`Measure`の値が表示される。尚、上記実施例では、2つのトラックに跨る演奏データが記憶されたセクタは、各トラック毎に2回ロードされているが、別のトラックで既にロードされている演奏データがどこかのキャッシュにあったら、その演奏データの再度のロードは行わず、あたかもそのキャッシュにロードしたかのようにしてロー

ドの時間を少なくするようにしてもよい。また上記実施例では、単純に、クロック数の少ないトラックの演奏データをロードするように構成されているが、空いているキャッシュにどのトラックの演奏データをロードするかを決定する処理（図18のステップ18\_3〜ステップ18\_8の処理）を変更して、ロード済み総クロック数だけではなく、ディスクのヘッド移動時間等も加味した評価により、どのトラックの演奏データをロードするかを決定してもよい。

【0079】

【発明の効果】以上説明したように、本発明の演奏データ読込装置は、空いている（または空いた）内部メモリにどのトラックの演奏データをロードするかを固定的には定めず、ロードされている演奏データの総クロックの最小のトラックの演奏データもしくはこれにディスク装置のアクセスのためのヘッド位置を加味したトラックの演奏データをロードするものであるため、内部メモリの利用効率が上がり、少ない数の内部メモリでクロックに正確な演奏が可能となる。

【図面の簡単な説明】

【図1】本発明の演奏データ読込装置の一実施例の構成を表わすブロック図である。

【図2】本実施例におけるデータ転送処理の基本概念を示したフローチャートである。

【図3】本実施例におけるデータ転送処理の基本概念を示したデータの流れを示す説明図である。

【図4】フロッピィディスク中の、今から演奏しようとするある曲の演奏データのデータ構造を示した図である。

【図5】データ転送のためのルーチンで使用する各種フラグ、レジスタ等の値の変化を示した模式図である。

【図6】データ転送のためのルーチンで使用する各種フラグ、レジスタ等の値の変化を示した模式図である。

【図7】データ転送のためのルーチンで使用する各種フラグ、レジスタ等の値の変化を示した模式図である。

【図8】データ転送のためのルーチンで使用する各種フラグ、レジスタ等の値の変化を示した模式図である。

【図9】データ転送のためのルーチンで使用する各種フラグ、レジスタ等の値の変化を示した模式図である。

【図10】データ転送のためのルーチンで使用する各種フラグ、レジスタ等の値の変化を示した模式図であ

る。

【図11】データ転送のためのルーチンで使用する各種フラグ、レジスタ等の値の変化を示した模式図である。

【図12】データ転送のためのルーチンで使用する各種フラグ、レジスタ等の値の変化を示した模式図である。

【図13】データ転送のためのルーチンで使用する各種フラグ、レジスタ等の値の変化を示した模式図である。

【図14】データ転送のためのルーチンで使用する各種フラグ、レジスタ等の値の変化を示した模式図である。

【図15】フロッピィディスクに記憶された演奏データのフォーマットを示した図である。

【図16】演奏データのフォーマットを示した図である。

【図17】演奏データ中のデータ情報部分の詳細を示した図である。

【図18】メインルーチンのフローチャートである。

【図19】Songで示される曲の演奏準備を行なうサブルーチンのフローチャートである。

【図20】次の演奏データをセットする処理を行なうサブルーチンのフローチャートである。

【図21】トラックの演奏データをロードするロードデータサブルーチンのフローチャートである。

【図22】クロック割込みルーチンのフローチャートである。

【図23】演奏データをその種類によりMIDI出力端子に出力したり、テンポ、拍子を更新したり、曲の終りを検出するサブルーチンのフローチャートである。

【図24】演奏停止処理を行なうサブルーチンのフローチャートである。

【図25】小節カウンタの曲頭の休符区間を検出し、楽譜通りの小節番号にするための処理を行なうサブルーチンのフローチャートである。

【図26】小節番号を表示する処理を行なうサブルーチンのフローチャートである。

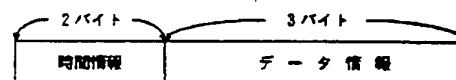
【図27】従来の手法における演奏データのロード方法を示すフローチャートである。

【図28】従来の手法の説明図である。

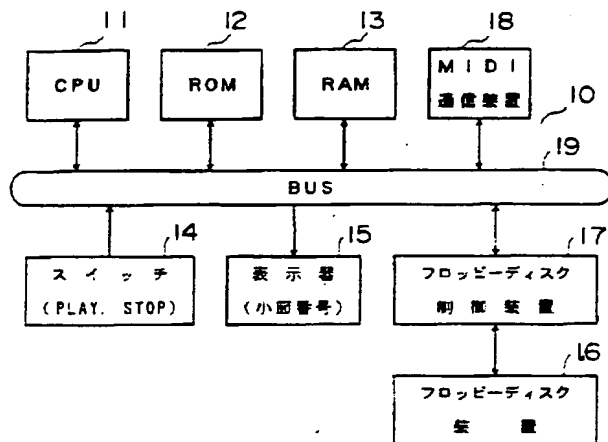
【図4】

トラック番号	track0		track1				track2		
そのセクタ内のクロック数	150	10	10	50	50	50	20	100	40
セクタ番号	sector0	sector1		sector2	sector3	sector4		sector5	sector6

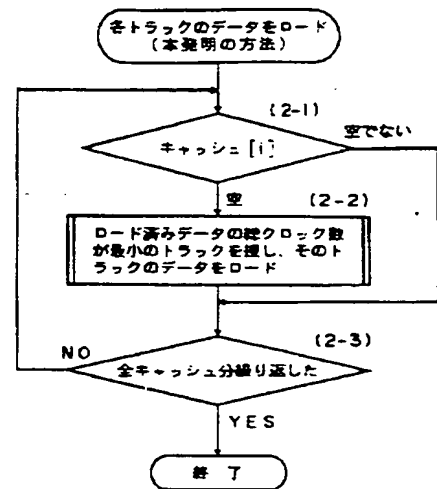
【図16】



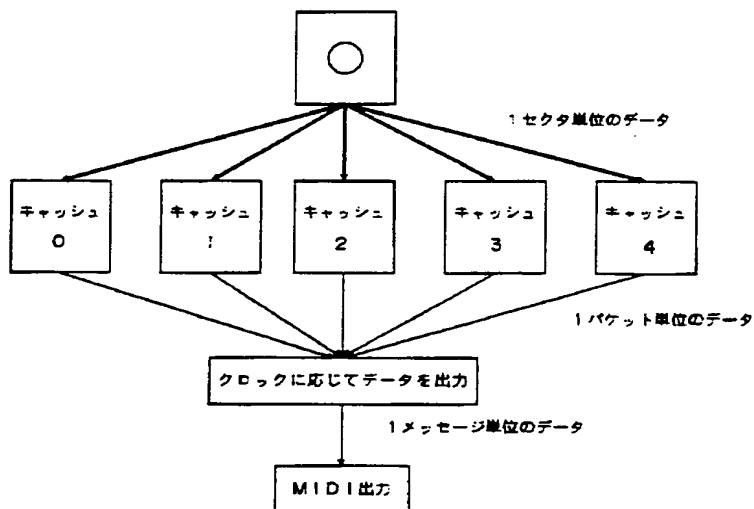
【図 1】



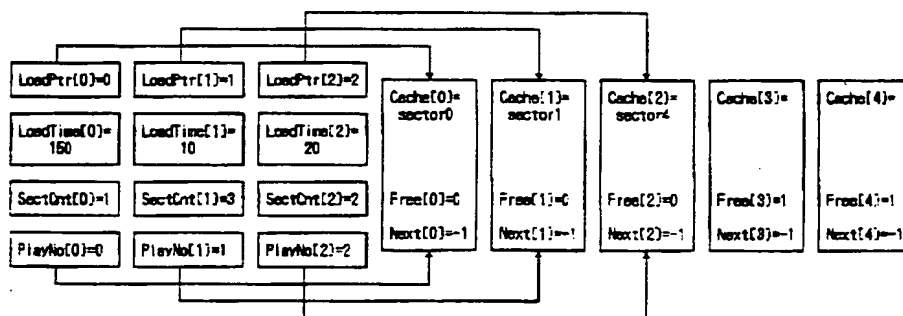
【図 2】



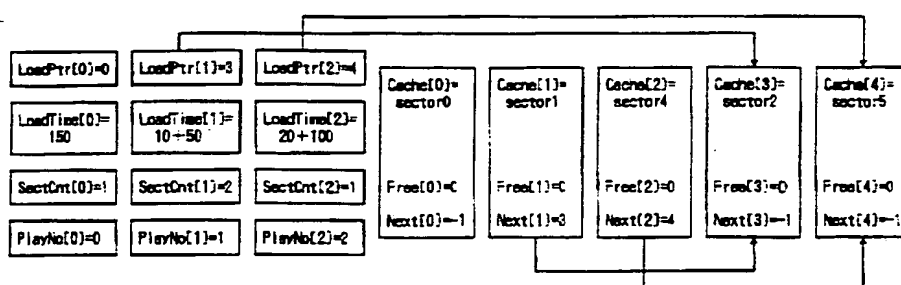
【図 3】



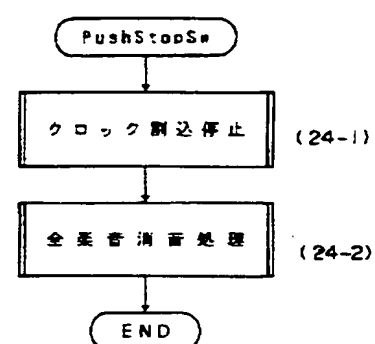
【図 5】



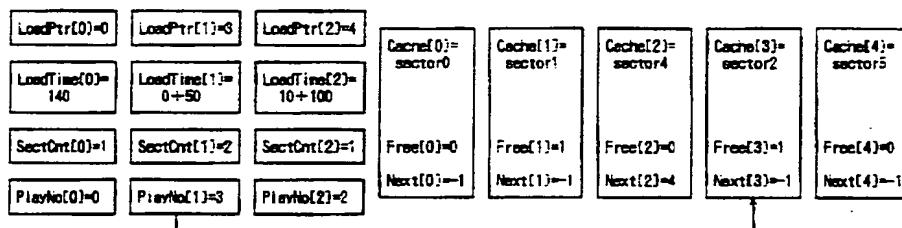
【図 6】



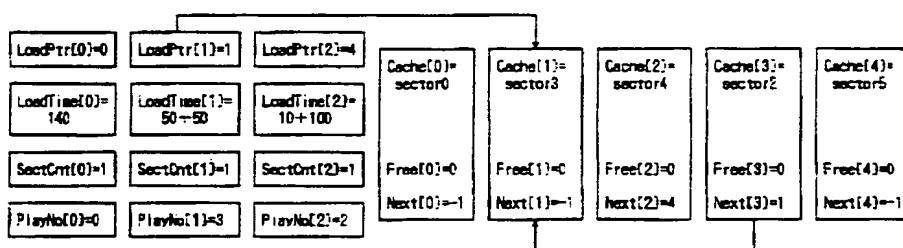
【図 24】



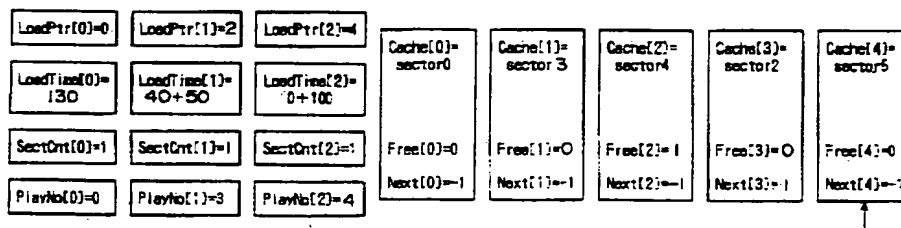
【図 7】



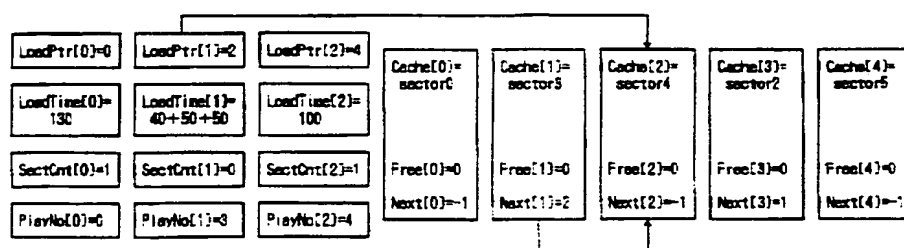
【図 8】



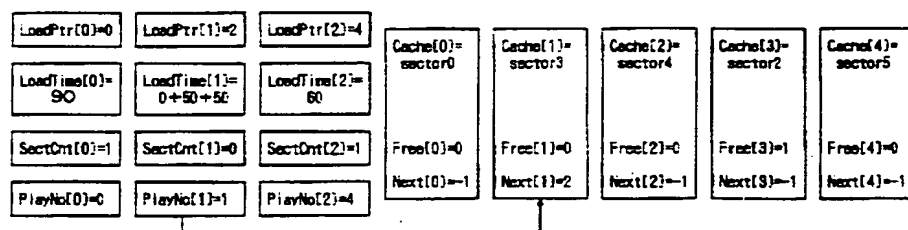
【図 9】



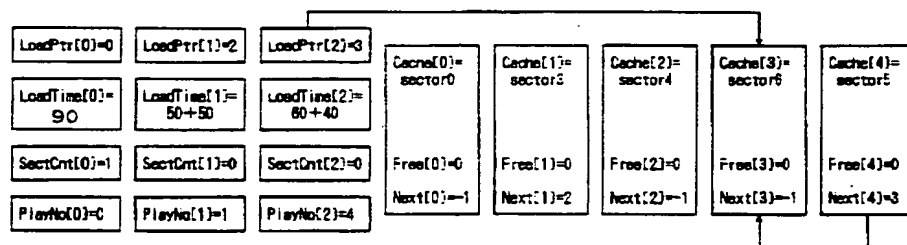
【図 1 0】



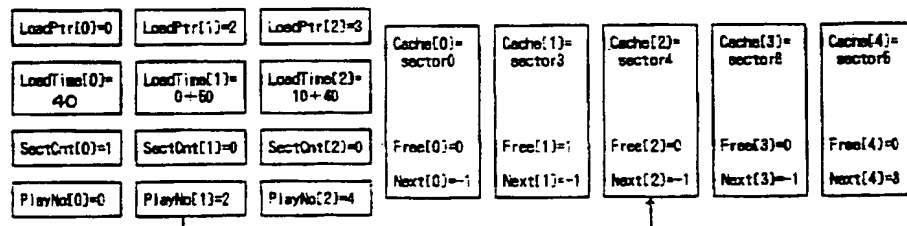
【図 1 1】



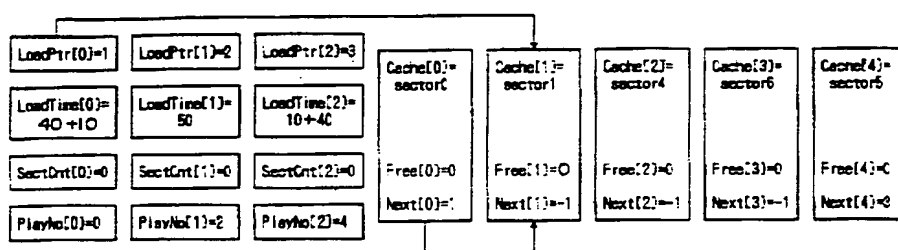
【図 1 2】



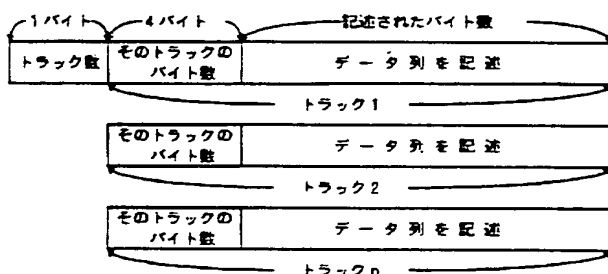
【図 1 3】



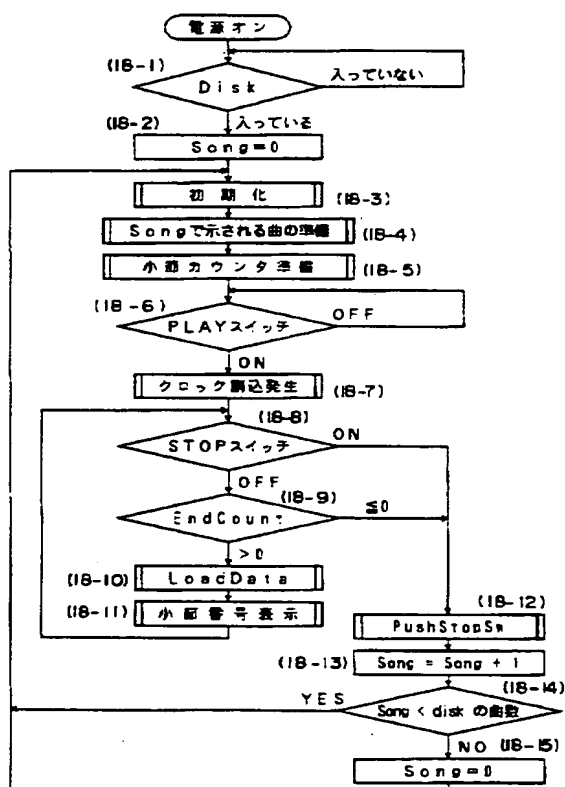
【図14】



【図15】



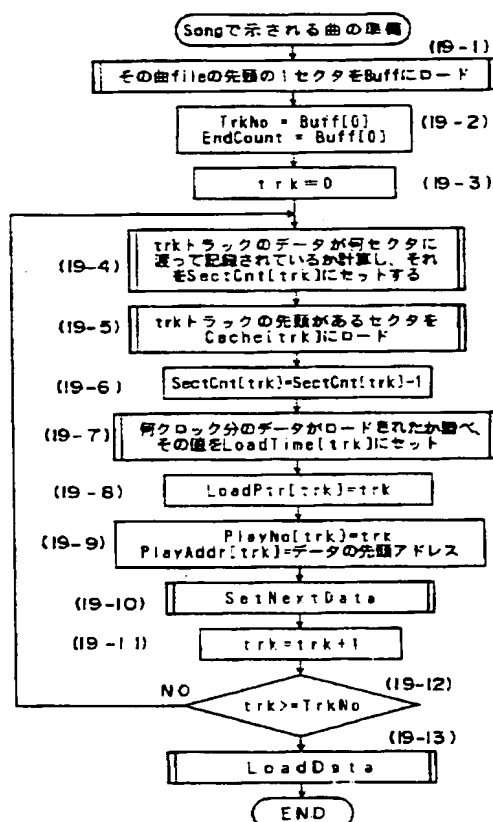
【図18】



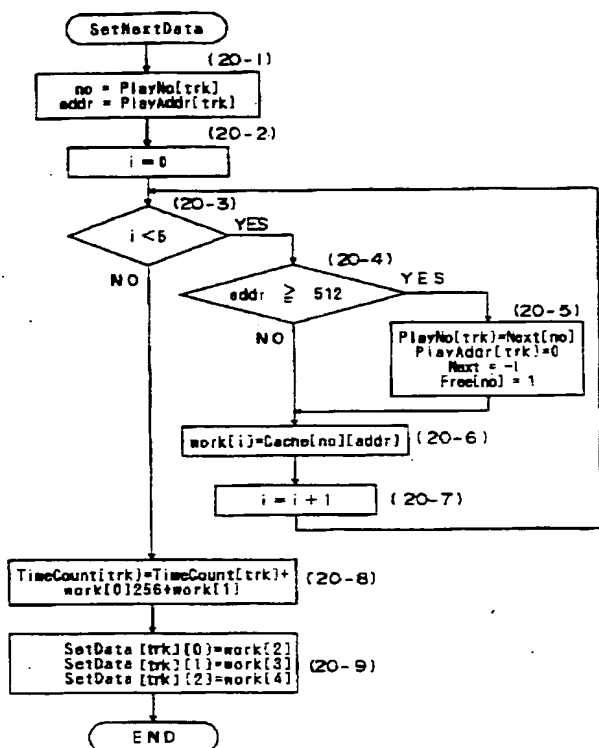
【図17】

MIDI情報 (3バイト)	時間情報	90h	3Ch	40h
MIDI情報 (2バイト)	時間情報	C0h	01h	FFh
MIDI情報 (1バイト)	時間情報	FFh	FFh	FFh
テンポ	時間情報	FFh	FFh	FFh
拍子	時間情報	F8h	00h	78h (テンポ120)
終了	時間情報	FEh	01h	40h (1/4)
	時間情報	FFh	FFh	FFh

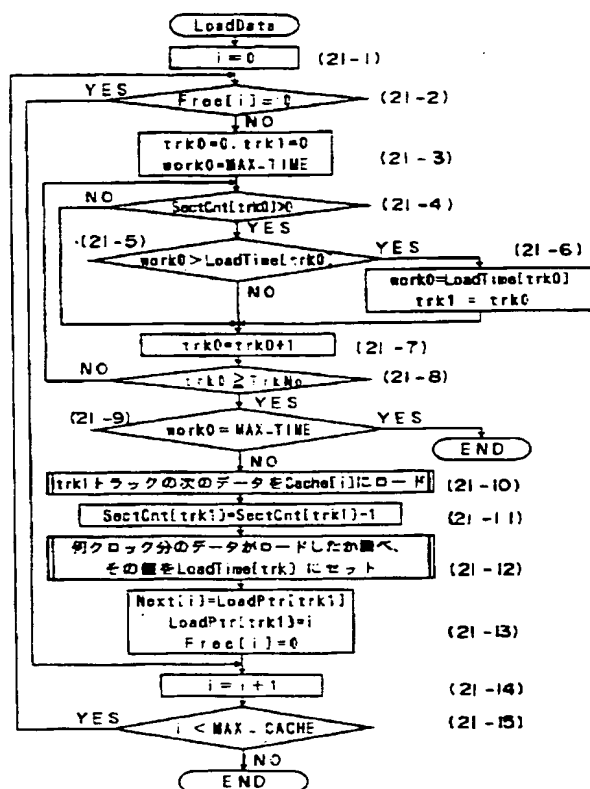
【図19】



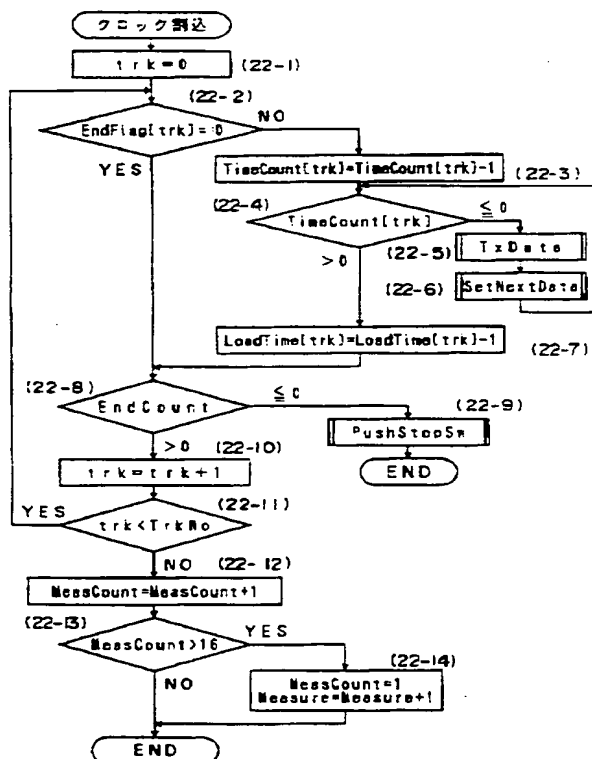
【図20】



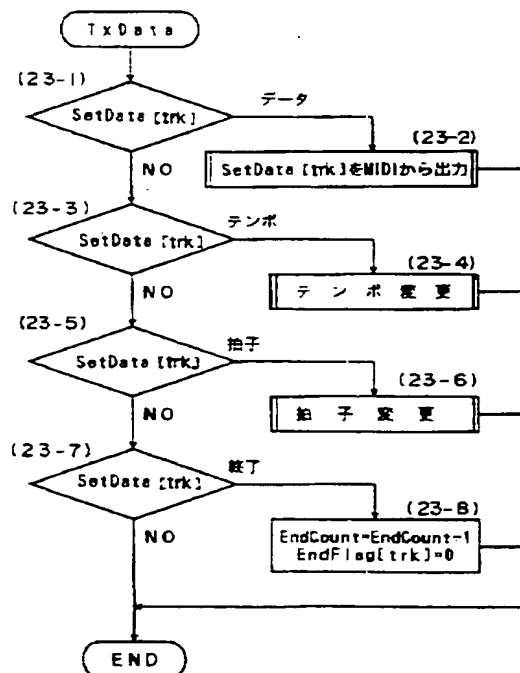
【図21】



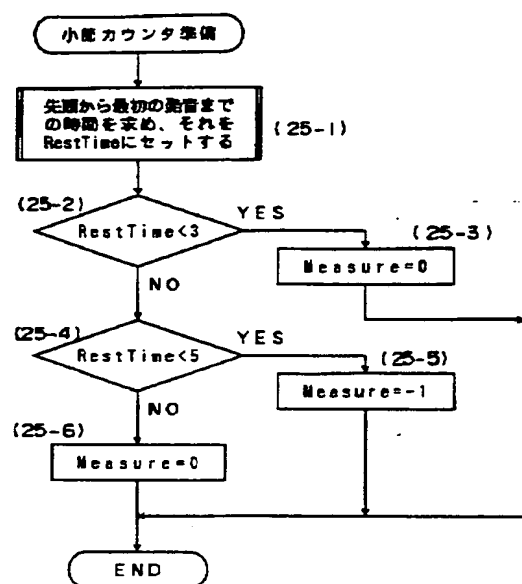
【図22】



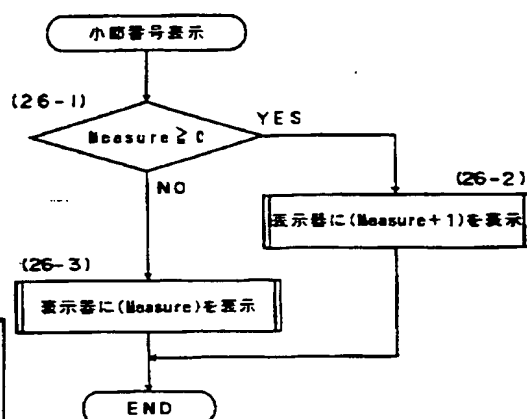
【図23】



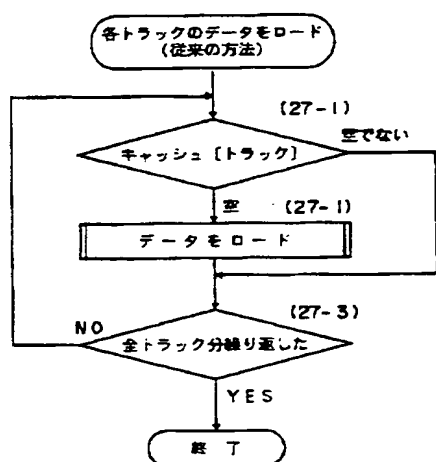
【図25】



【図26】



【図27】



【図28】

